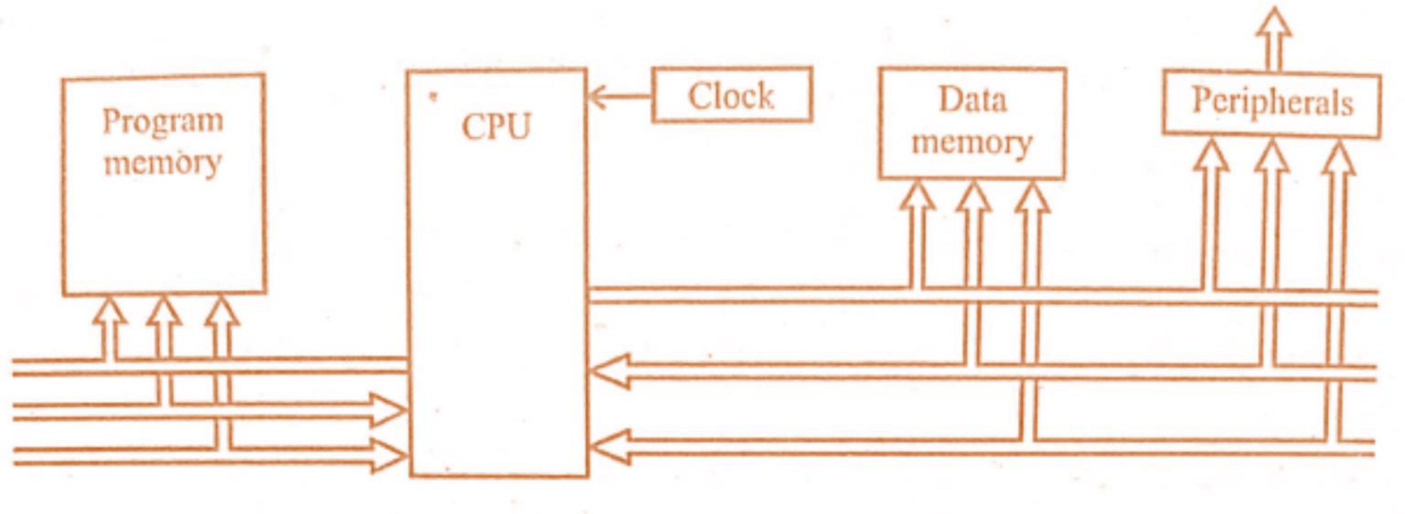
Digital Signal Processors

- The programmable digital signal processors (PDSPs) are general purpose microprocessors designed specifically for digital signal processing applications.
- They contain special architecture and instruction set to execute computation - intensive DSP algorithms more efficiently.
  - ***General purpose digital signal processors***: These are basically high-speed microprocessors with architecture and instruction sets optimized for DSP operations.
  - ***Special purpose digital signal processors:*** These types of processors consist of hardware i) designed for specific DSP algorithms such as FFr, ii) hardware designed for specific applications such as PCM and filtering.

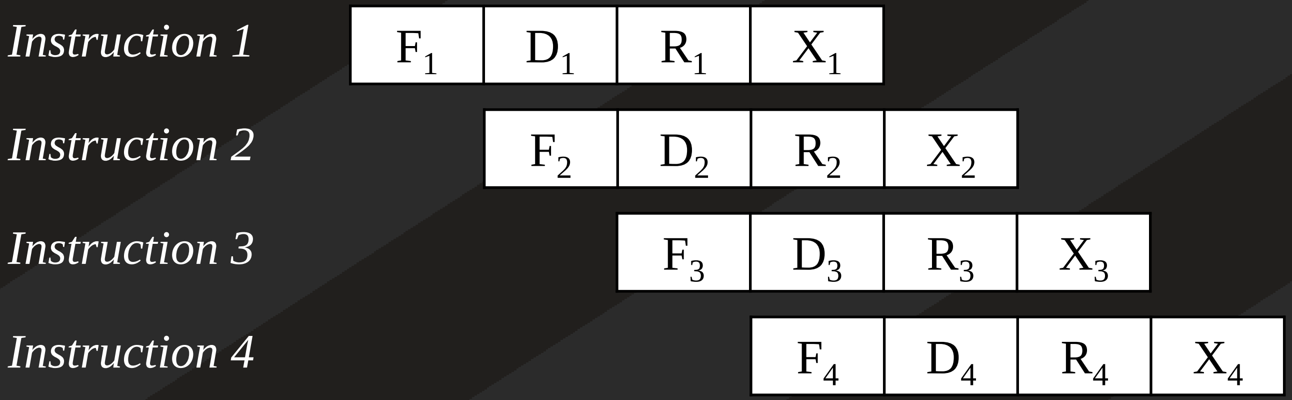
Harvard Architecture

- The term Harvard originated from the Harvard Mark 1 relay-based computer which stored instruction on punched tape and data in relay latches
- The Harvard architectures physically separate memories for their instructions and data, requiring dedicated buses for each of them.
- Instructions and operands can therefore be fetched simultaneously.
- Most of the DSP processors use a modified Harvard architecture with two or three memory buses; allowing access to filter coefficients and input signals in the same cycle.
- Since it possesses two independent bus systems, the Harvard architecture is capable of simultaneous reading an instruction code and reading or writing a memory or peripheral as part of the execution of the previous instruction.



Pipelining

- To improve the efficiency, advanced microprocessors and digital signal processors use an approach called pipelining in which different phases of operation and execution of instructions are carried out in parallel.
- In modem processors the first step of execution is performed on the first instruction, and then when the instruction passes to the next step, a new instruction is started.
    - ***The Fetch phase(F)*** in which the next instruction is fetched from the address stored in the program counter.
    - ***The decode phase (D)*** in which the instruction in the instruction register is decoded and the address in the program counter is incremented
    - ***Memory read (R)*** phase reads the data from the data buses and also writes data to the data buses.
    - ***The Execute phase (X)*** executes the instruction currently in the instruction register and also completes the write process.



Instruction 1	F_1	D_1	R_1	X_1			
Instruction 2		F_2	D_2	R_2	X_2		
Instruction 3			F_3	D_3	R_3	X_3	
Instruction 4				F_4	D_4	R_4	X_4

Multiply Accumulate Unit (MAC)

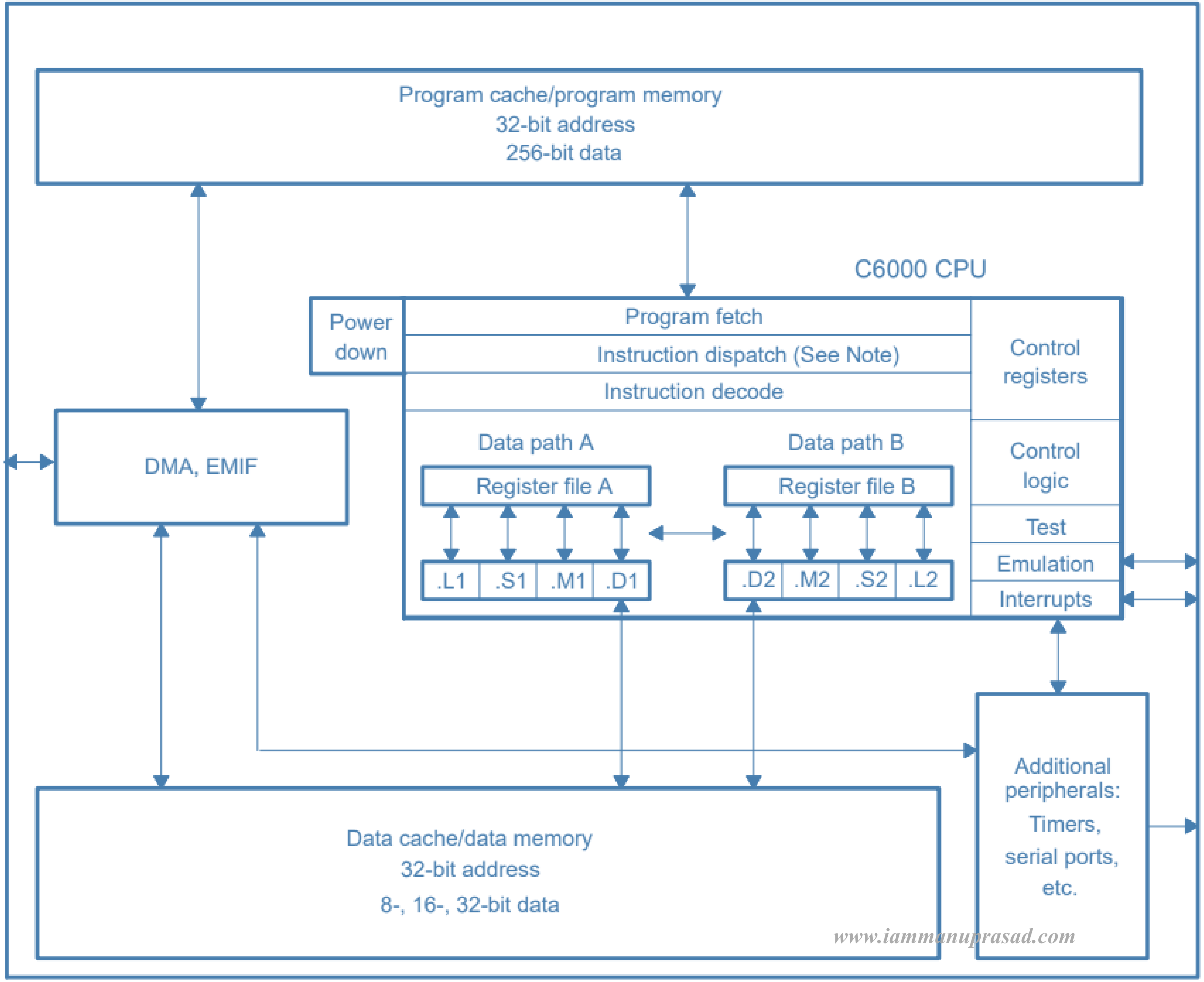
- The Multiply-Accumulate (MAC) operation is the basis of many digital signal processing algorithms
- In digital signal processing, the multiply–accumulate (MAC) operation is a common step that computes the product of two numbers and adds that product to an accumulator.
- The hardware unit that performs the operation is known as a multiplier–accumulator (MAC unit); the operation itself is also often called a MAC
- The MAC speed applies both to finite impulse response (FIR) and infinite impulse response (IIR) fi1ters. The complexity of the filter response dictates the number MAC operations required per sample period.
- A multiply-accumulate step performs the following:
  - Reads a 16-bit sample data (pointed to by a register)
  - Increments the sample data-pointer by 2
  - Reads a. 16-bit coefficient (pointed to by another register)
  - Increments the coefficient register pointer by 2
  - Sign Multiply (16-bit) data and coefficient 'to yield a 32~bit resu1t
  - Adds the result to the contents of a 32-bit register pair for accumulate.

TMS320C67xx - Digital Signal Processor

- The **TMS320 DSP** family consists of fixed-point, floating-point, and multiprocessor digital signal processors (DSPs).
- TMS320 DSPs have an architecture designed specifically for real-time signal processing.
- With a performance of up to 6000 million instructions per second (MIPS) and an efficient C compiler, the **TMS320C6000 DSPs** give system architects unlimited possibilities to differentiate their products.
  - High Performance
  - Ease of use
  - affordable pricing
- The C6000 devices execute up to eight 32-bit instructions per cycle. The C67x CPU consists of 32 general-purpose 32-bit registers and eight functional units.
- These eight functional units contain:
  - Two multipliers
  - Six ALUs

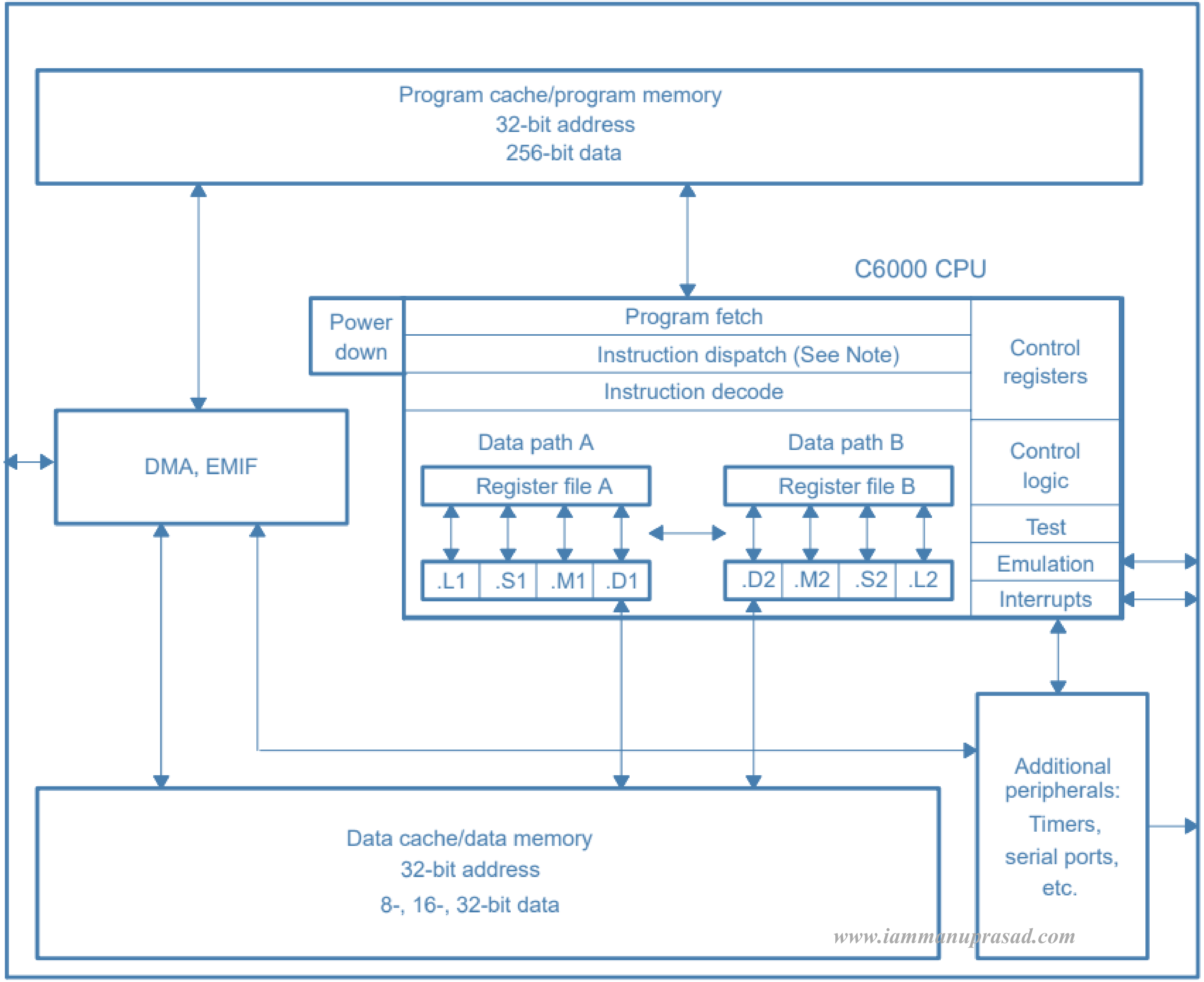
TMS320C67xx DSP Architecture

- Central Processing Unit (CPU)
- Internal Memory
- Memory and Peripheral Options



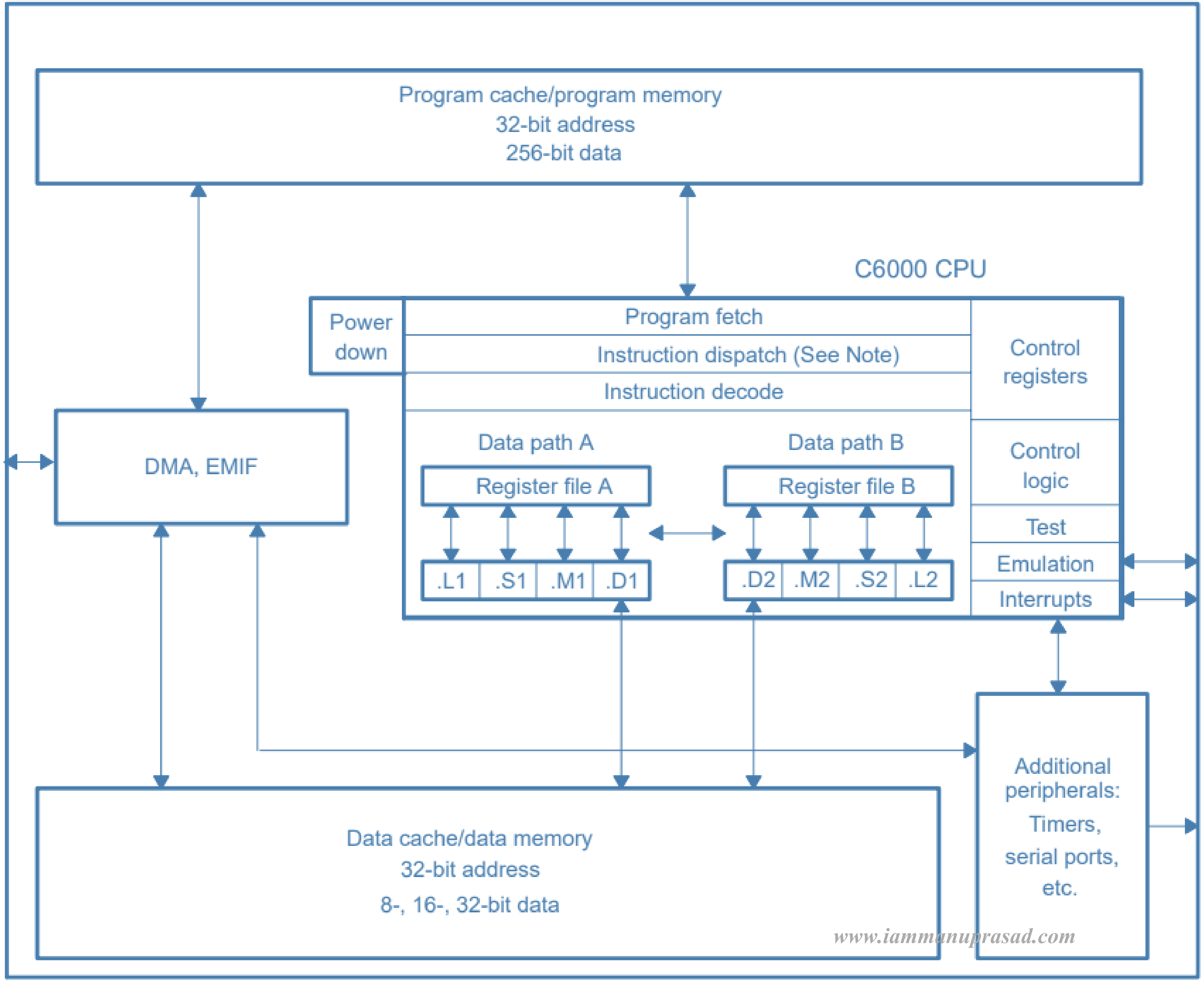
TMS320C67xx DSP Architecture

- **Central Processing Unit (CPU)**
  - Program fetch unit
  - Instruction dispatch unit
  - Instruction decode unit
  - Two data paths, each with four functional units
  - 32 32-bit registers
  - Control registers
  - Control logic
  - Test, emulation, and interrupt logic



TMS320C67xx DSP Architecture

- **DMA** – Direct memory Access
- **EMIF** – External Memory interface
- Program memory/Program cache
- Data memory /Data cache



TMS320C67xx DSP Architecture

- DMA Controller (C6701 DSP only) transfers data between address ranges in the memory map without intervention by the CPU. The DMA controller has four programmable channels and a fifth auxiliary channel.
- EDMA Controller performs the same functions as the DMA controller. The EDMA has 16 programmable channels, as well as a RAM space to hold multiple configurations for future transfers.
- HPI is a parallel port through which a host processor can directly access the CPU's memory space. The host device has ease of access because it is the master of the interface. The host and the CPU can exchange information via internal or external memory. In addition, the host has direct access to memory-mapped peripherals.
- Expansion bus is a replacement for the HPI, as well as an expansion of the EMIF. The expansion provides two distinct areas of functionality (host port and I/O port) which can co-exist in a system. The host port of the expansion bus can operate in either asynchronous slave mode, similar to the HPI, or in synchronous master/slave mode. This allows the device to interface to a variety of host bus protocols. Synchronous FIFOs and asynchronous peripheral I/O devices may interface to the expansion bus.

TMS320C67xx DSP Architecture

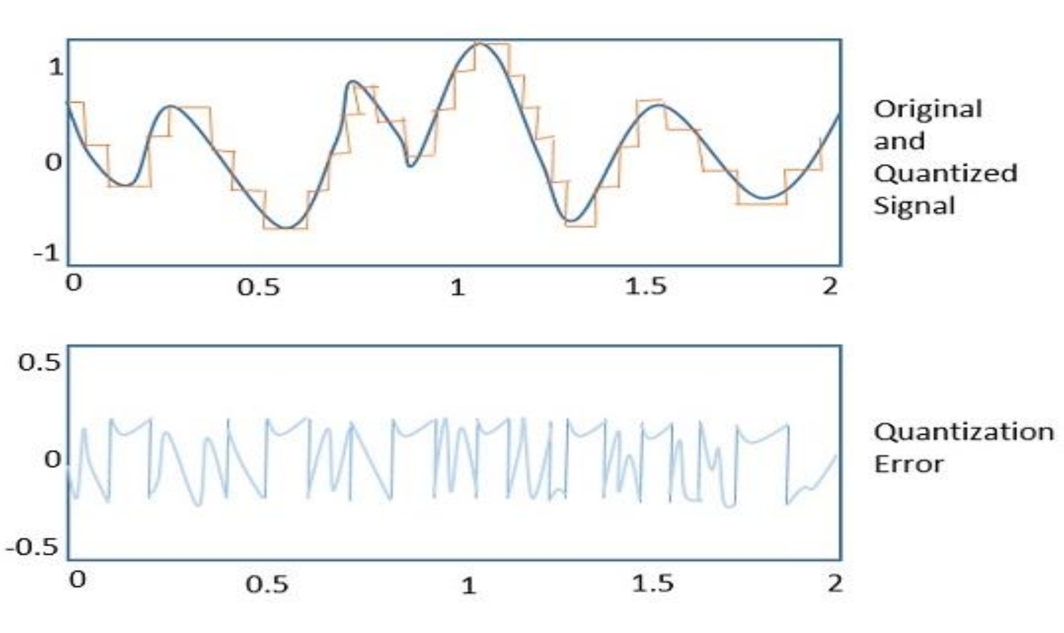
- McBSP (multichannel buffered serial port) is based on the standard serial port interface found on the TMS320C2000 and TMS320C5000 devices. In addition, the port can buffer serial samples in memory automatically with the aid of the DMA/EDNA controller. It also has multichannel capability compatible with the T1, E1, SCSA, and MVIP networking standards.
- Timers in the C6000 devices are two 32-bit general-purpose timers used for these functions:
  - Time events
  - Count events
  - Generate pulses
  - Interrupt the CPU
  - Send synchronization events to the DMA/EDMA controller.
- Power-down logic allows reduced clocking to reduce power consumption. Most of the operating power of CMOS logic dissipates during circuit switching from one logic state to another. By preventing some or all of the chip’s logic from switching, you can realize significant power savings without losing any data or operational context.

Finite Word length Effects

- In the design of FIR Filters, the filter coefficients are determined by the system transfer functions. These filters co-efficient are quantized/truncated while implementing DSP System because of finite length registers.
- Only Finite numbers of bits are used to perform arithmetic operations. Typical word length is 16 bits, 24 bits, 32 bits etc.
- This finite word length introduces an error which can affect the performance of the DSP system.
  - Input quantization error
  - Co-efficient quantization error
  - Overflow & round off error (Product Quantization error)

Quantization Error

- The effect of error introduced by a signal process depend upon number of factors including the.
  - Type of arithmetic
  - Quality of input signal
  - Type of algorithm implemented
- For any system, during its functioning, there is always a difference in the values of its input and output. The processing of the system results in an error, which is the difference of those values. The difference between an input value and its quantized value is called a Quantization Error.



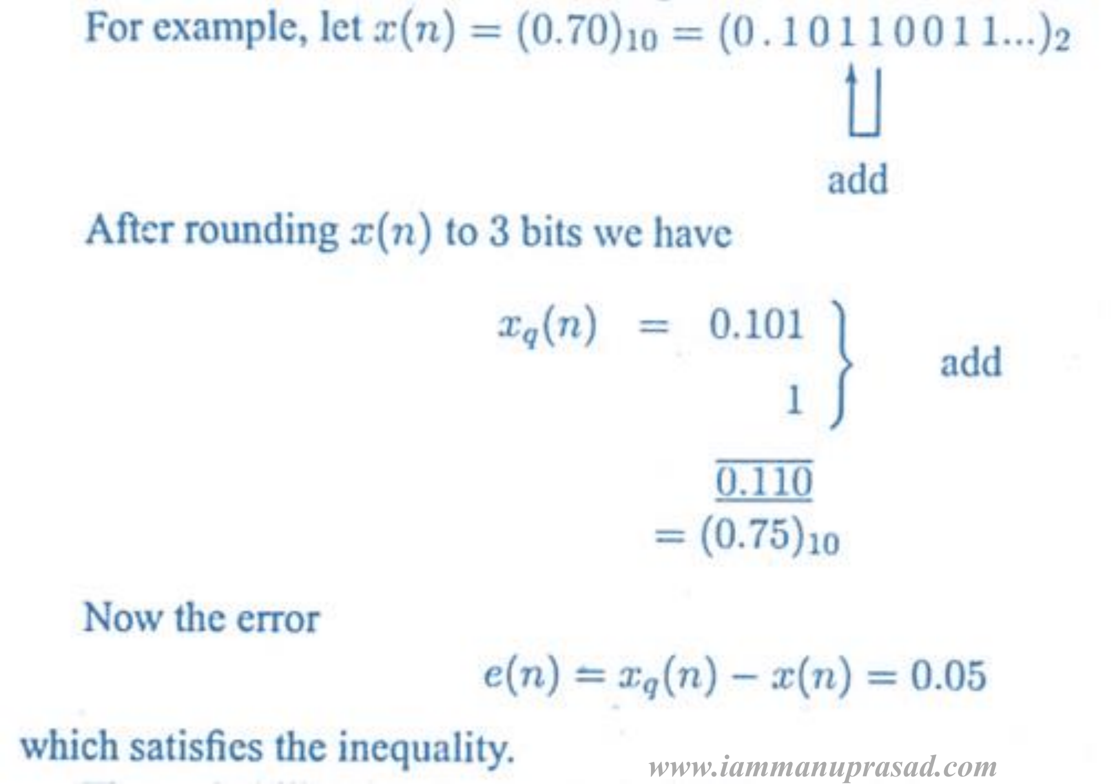
Input quantization error

- The conversion of continuous-time input signal into digital value produces an error which is known as input quantization error. This error arises due to the representation of the input signal by a fixed number of digits in A/D conversion process

$$e(n) = x_q(n) - x(n)$$

$x_q(n) \rightarrow sample\ quatised\ value$

$x(n) \rightarrow input\ value$



For example, let $x(n) = (0.70)_{10} = (0.10110011...)_2$

add

After rounding $x(n)$ to 3 bits we have

$$\left.\begin{aligned} x_q(n) &= 0.101 \\ &1 \end{aligned}\right\} \text{add}$$

$$\overline{0.110}$$

$$= (0.75)_{10}$$

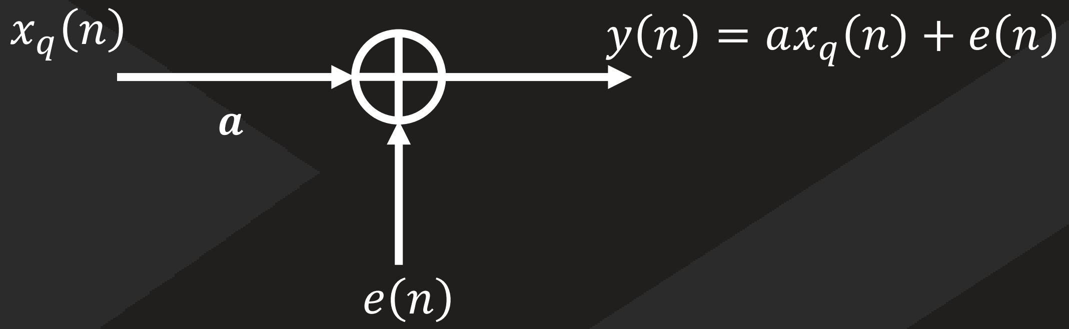
Now the error

$$e(n) = x_q(n) - x(n) = 0.05$$

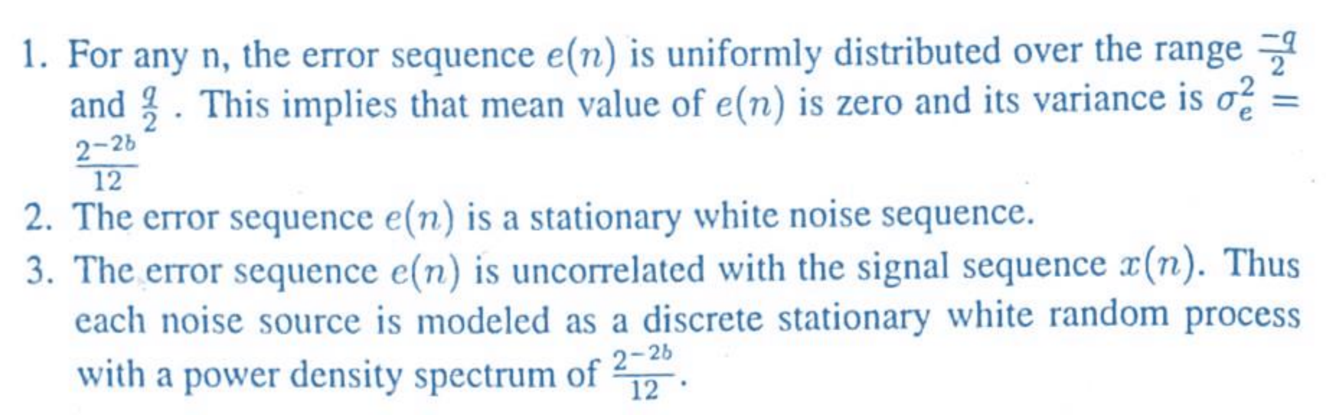
which satisfies the inequality.

Product Quantization error

- In fixed point arithmetic the product of two *b-bit* numbers results in *2b* bits long. In DSP applications it is necessary to round this product to *b-bit* number which produce an error known as product quantization error or product round off noise



- The multiplication is modelled as an infinite precision multiplier followed by an adder where round off noise is added to the product so that overall result equals some quantization level



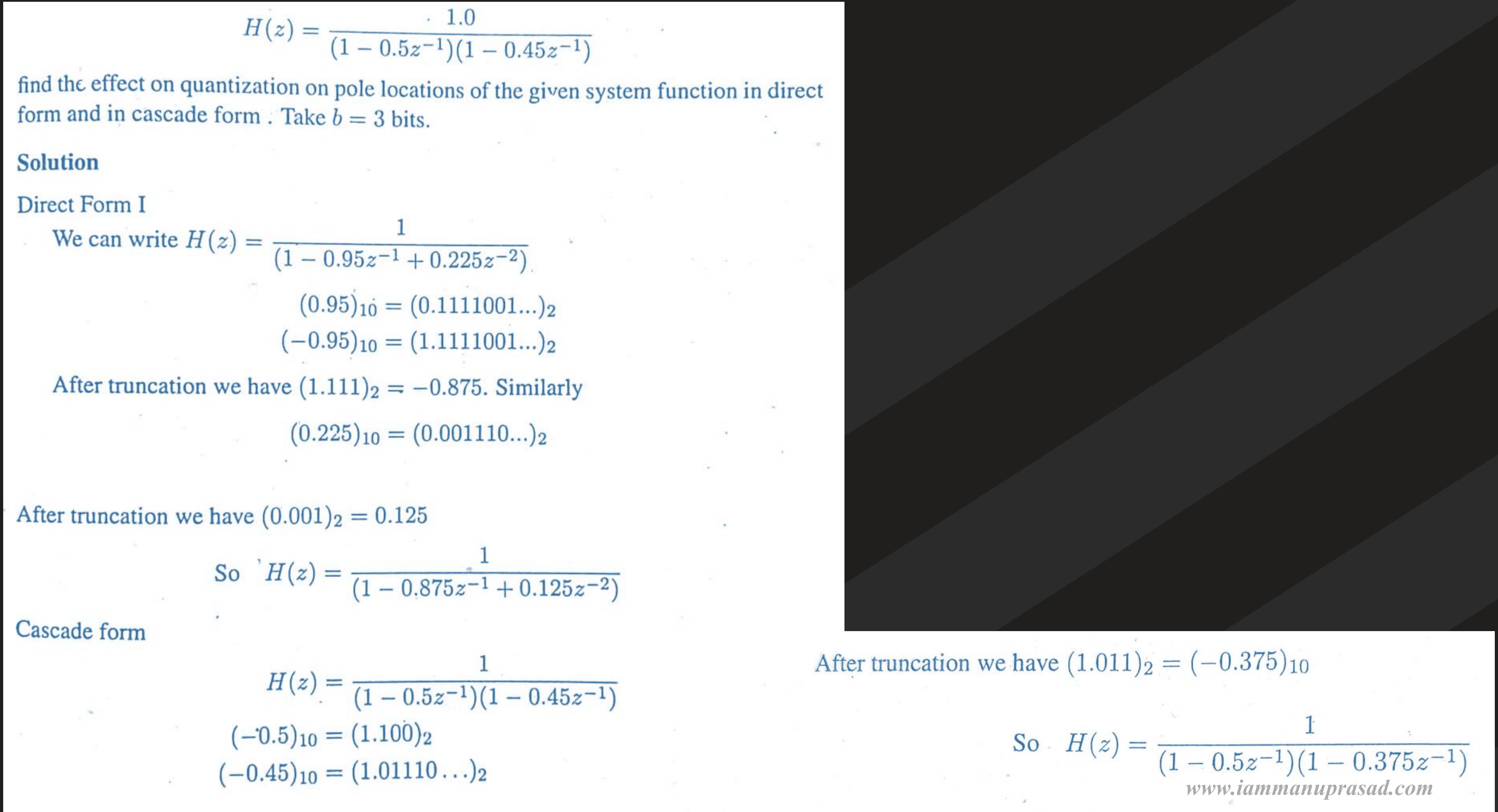
1. For any n, the error sequence $e(n)$ is uniformly distributed over the range $\frac{-q}{2}$ and $\frac{q}{2}$. This implies that mean value of $e(n)$ is zero and its variance is $\sigma_e^2 = \frac{2^{-2b}}{12}$
2. The error sequence $e(n)$ is a stationary white noise sequence.
3. The error sequence $e(n)$ is uncorrelated with the signal sequence $x(n)$. Thus each noise source is modeled as a discrete stationary white random process with a power density spectrum of $\frac{2^{-2b}}{12}$.

Coefficient quantization error

- In the design of a digital filter the coefficients are evaluated with infinite precision.
- But when they are quantized, the frequency response of the actual filter deviates from that which would have been obtained with an infinite word length representation and the filter may actually fail to meet the desired specifications.
- If the poles of the desired filter are close to the unit circle, then those of the filter with quantized coefficients my lie just outside the unit circle

Coefficient quantization error

- Consider a second order IIR filter with



$$H(z) = \frac{1.0}{(1-0.5z^{-1})(1-0.45z^{-1})}$$

find the effect on quantization on pole locations of the given system function in direct form and in cascade form . Take $b = 3$ bits.

Solution

Direct Form I

We can write $H(z) = \dfrac{1}{(1-0.95z^{-1}+0.225z^{-2})}$

$$(0.95)_{10} = (0.1111001...)_2$$
$$(-0.95)_{10} = (1.1111001...)_2$$

After truncation we have $(1.111)_2 = -0.875$. Similarly

$$(0.225)_{10} = (0.001110...)_2$$

After truncation we have $(0.001)_2 = 0.125$

So $H(z) = \dfrac{1}{(1-0.875z^{-1}+0.125z^{-2})}$

Cascade form

$$H(z) = \frac{1}{(1-0.5z^{-1})(1-0.45z^{-1})}$$
$$(-0.5)_{10} = (1.100)_2$$
$$(-0.45)_{10} = (1.01110\ldots)_2$$

After truncation we have $(1.011)_2 = (-0.375)_{10}$

So $H(z) = \dfrac{1}{(1-0.5z^{-1})(1-0.375z^{-1})}$